JAVA.UTIL.TREEMAP CLASS

Introduction

The **java.util.TreeMap** class is the Red-Black tree based implementation of the Map interface.Following are the important points about TreeMap:

- The TreeMap class guarantees that the Map will be in ascending key order.
- The Map is sorted according to the natural sort method for the key Class, or by the Comparator provided at map creation time, that will depend on which constructor used.

Class declaration

Following is the declaration for **java.util.TreeMap** class:

```
public class TreeMap<K,V>
   extends AbstractMap<K,V>
      implements NavigableMap<K,V>, Cloneable, Serializable
```

Parameters

Following is the parameter for **java.util.TreeMap** class:

- **K** -- This is the type of keys maintained by this map.
- **V** -- This is the type of mapped values.

Class constructors

S.N.	Constructor & Description
1	**TreeMap()** This constructor constructs a new, empty tree map, using the natural ordering of its keys.
2	**TreeMap(Comparator<? super K> comparator)** This constructor constructs a new, empty tree map, ordered according to the given comparator.
3	**TreeMap(Map<? extends K,? extends V> m)** This constructor constructs a new tree map containing the same mappings as the given map, ordered according to the natural ordering of its keys.
4	**TreeMap(SortedMap<K,? extends V> m)** This constructor constructs a new tree map containing the same mappings and using the same ordering as the specified sorted map.

Class methods

S.N.	Method & Description

1	Map.Entry<K,V> ceilingEntry(K key) This method returns a key-value mapping associated with the least key greater than or equal to the given key, or null if there is no such key.
2	K ceilingKey(K key) This method returns the least key greater than or equal to the given key, or null if there is no such key.
3	void clear() This method removes all of the mappings from this map.
4	Object clone() This method returns a shallow copy of this TreeMap instance.
5	Comparator<? super K> comparator() This method Returns the comparator used to order the keys in this map, or null if this map uses the natural ordering of its keys.
6	boolean containsKey(Object key) This method returns true if this map contains a mapping for the specified key.
7	boolean containsValue(Object value) This method returns true if this map maps one or more keys to the specified value.
8	NavigableSet<K> descendingKeySet() This method returns a reverse order NavigableSet view of the keys contained in this map.
9	NavigableMap<K,V> descendingMap() This method returns a reverse order view of the mappings contained in this map.
10	Set<Map.Entry<K,V>> entrySet() This method returns a Set view of the mappings contained in this map.
11	Map.Entry<K,V> firstEntry() This method returns a key-value mapping associated with the least key in this map, or null if the map is empty.
12	K firstKey() This method returns the first (lowest) key currently in this map.
13	Map.Entry<K,V> floorEntry(K key) This method returns a key-value mapping associated with the greatest key less than or equal to the given key, or null if there is no such key.
14	K floorKey(K key) This method returns the greatest key less than or equal to the given key, or null if there is no such key.
15	V get(Object key) This method returns the value to which the specified key is mapped, or null if this map contains no mapping for the key.
16	SortedMap<K,V> headMap(K toKey) This method returns a view of the portion of this map whose keys are strictly less than toKey.
17	NavigableMap<K,V> headMap(K toKey, boolean inclusive) This method returns a view of the portion of this map whose keys are less than (or equal to, if inclusive is true) toKey.

18	Map.Entry<K,V> higherEntry(K key) This method returns the returns a key-value mapping associated with the least key strictly greater than the given key, or null if there is no such key.
19	K higherKey(K key) This method returns the least key strictly greater than the given key, or null if there is no such key.
20	Set<K> keySet() This method returns a Set view of the keys contained in this map.
21	Map.Entry<K,V> lastEntry() This method returns a key-value mapping associated with the greatest key in this map, or null if the map is empty.
22	K lastKey() This method returns the last (highest) key currently in this map.
23	Map.Entry<K,V> lowerEntry(K key) This method returns a key-value mapping associated with the greatest key strictly less than the given key, or null if there is no such key.
24	K lowerKey(K key) This method returns the greatest key strictly less than the given key, or null if there is no such key.
25	NavigableSet<K> navigableKeySet() This method returns a NavigableSet view of the keys contained in this map.
26	Map.Entry<K,V> pollFirstEntry() This method removes and returns a key-value mapping associated with the least key in this map, or null if the map is empty.
27	Map.Entry<K,V> pollLastEntry() This method removes and returns a key-value mapping associated with the greatest key in this map, or null if the map is empty.
28	V put(K key, V value) This method associates the specified value with the specified key in this map.
29	void putAll(Map<? extends K,? extends V> map) This method copies all of the mappings from the specified map to this map.
30	V remove(Object key) This method removes the mapping for this key from this TreeMap if present.
31	int size() This method returns the number of key-value mappings in this map.
32	NavigableMap<K,V> subMap(K fromKey, boolean fromInclusive, K toKey, boolean toInclusive) This method returns a view of the portion of this map whose keys range from fromKey to toKey
33	SortedMap<K,V> subMap(K fromKey, K toKey) This method returns a view of the portion of this map whose keys range from fromKey, inclusive, to toKey, exclusive
34	SortedMap<K,V> tailMap(K fromKey) This method returns a view of the portion of this map whose keys are greater than or equal to fromKey.

35	NavigableMap<K,V> tailMap(K fromKey, boolean inclusive) This method returns a view of the portion of this map whose keys are greater than (or equal to, if inclusive is true) fromKey.
36	Collection<V> values() This method returns a Collection view of the values contained in this map.

Methods inherited

This class inherits methods from the following classes:

- java.util.AbstractMap
- java.util.Object
- java.util.Map